

# The Visualization of Change in Word Meaning over Time using Temporal Word Embeddings

**Chiraag Lala**

Out of The View Ventures

London, UK

chiraag.r.lala@gmail.com

**Shay B. Cohen**

School of Informatics

University of Edinburgh, Edinburgh, UK

scohen@inf.ed.ac.uk

## Abstract

We describe a visualization tool that can be used to view the change in meaning of words over time. The tool makes use of existing (static) word embedding datasets together with a timestamped  $n$ -gram corpus to create *temporal* word embeddings.

## 1 Introduction

Embedding words into a vector space and using these vectors in various applications has been a recent topic of great interest in NLP (Turian et al., 2010; Dhillon et al., 2011; Collobert et al., 2011; Mikolov et al., 2011; Blacoe and Lapata, 2012; Faruqui and Dyer, 2014). Many of these methods rely on dimensionality reduction techniques for vectors which represent distributional features of the words.

In this short paper, we describe an approach to visualize the change in meaning of words over time using such word embeddings. The main idea is to embed words, given as a query, into a vector space, in a running sequence of time slices. Once these embeddings are calculated, they can be represented on a 2D plane. The points in the 2D plane move in the space as the time slices change.

Assuming that word vectors close to each other in Euclidean space are semantically related, and assuming that distributional similarity and context of appearance of words also greatly determines the meaning of the word (Firth, 1961), our visualization tool can be viewed as a tool to inspect the change in word meaning over time.

## 2 Temporal Word Embeddings

Let  $V$  be a vocabulary – a set of words over some alphabet. We also assume a special symbol  $\perp \in V$ , which denotes an unknown

word token. For example, in our online visualization tool,  $V$  is a subset of the words that appear in the Google books  $n$ -gram corpus (Michel et al., 2011). We define a static word embedding function to be function  $f: V \rightarrow \mathbb{R}^d$ , which maps every word in  $V$  to a vector in some  $d$ -dimensional space. We experimented with several such word embedding functions, including those by Collobert et al. (2011) that make use of neural networks with the SENNA toolkit, the HLBL embeddings of Turian et al. (2010) and Mnih and Hinton (2007) and also the embeddings by Mikolov et al. (2011) that make use of the RNNLM toolkit.

Let  $T$  be a set of time slices. Each member  $t \in T$  denotes a certain span of time, for example, the years between 1810 and 1815. A temporal word embedding function for  $T$  and  $V$  is then a function  $g: T \times V \rightarrow \mathbb{R}^\ell$ . In our visualization tool,  $T$  denotes the span of years between 1800 and 2008, broken into 5 years intervals. We assume a function `TimeSlice` that maps an arbitrary timestamp  $t$  into its corresponding time slice in  $T$ . For example, in our experiments `TimeSlice(1801)` would map the timestamp 1801 to the range of years 1800-1805.

We now show how to convert a static word embedding function to a temporal word embedding function. We first assume an associative commutative operator  $\odot$  which operates on a pair of vectors. This operator takes as input a pair of vectors (not necessarily of the same dimension) and returns a new vector. In our visualization tool,  $\odot$  is the concatenation operator between vectors, and also the sum of vectors. In the latter case, we assume that the operator  $\odot$  always accepts vectors of the same length.

In order to create the temporal word embedding, we assume the existence of a dataset  $D$ . Each datum  $d \in D$  is an  $n$ -gram ( $n = 5$  in our case and in general should be odd, so a mid-

dle word can be identified), together with a timestamp  $t$  in which this  $n$ -gram appeared and a count for it  $c$  in a corpus from that time. Therefore,  $d = \langle w_1, \dots, w_n, t, c \rangle$ .

In order to convert  $f$  to a temporal word embedding  $g$ , we define  $g(t, w)$  to be:

$$g(t, w) = \sum_{\substack{d=\langle w_1, \dots, w_n, t', c \rangle \in D \\ \text{TimeSlice}(t')=t, w_{(n+1)/2}=w}} (c/N_{t,w}) \times (\bigodot_{i \neq (n+1)/2} f(w_i)), \quad (1)$$

where  $N_{t,w}$  is defined as:

$$N_{t,w} = \sum_{\substack{d=\langle w_1, \dots, w_n, t', c \rangle \in D \\ \text{TimeSlice}(t')=t, w_{(n+1)/2}=w}} c \quad (2)$$

This means that  $g(t, w)$  is calculated by running the operator  $\odot$  over all contexts of the word  $w$  in  $D$  (meaning, words to left and words to right) in time slice  $t$ .

**Data** For the development of the temporal word embeddings, we used a subset of the Google books 5-gram data (Michel et al., 2011) for years 1800-2008.

### 3 Multidimensional Scaling

Once we obtain the temporal function  $g(t, w)$ , we use it to embed a collection of words in a query into the plane, and have a smooth animation that moves the words through the time slices.

For a given query  $w_1, \dots, w_k$ , we compute  $g(t, w_i)$  for all  $i \in \{1, \dots, k\}$  and  $t \in T$ . Let  $v_{t,i} = g(t, w_i)$ . We then create a matrix  $A$  of size  $k|T| \times k|T|$ , where

$$A_{(i,t),(j,t')} = \|v_{t,i} - v_{t',j}\|_2.$$

The matrix  $A$  is therefore the distance matrix for all vectors across all time slices for all words in the query.

We then use multidimensional scaling with the distance matrix  $A$  to embed all points  $v_{t,i}$  in a 2D plane. Multidimensional scaling works by solving the following optimization problem over the variables  $x_{t,i} \in \mathbb{R}^2$  for  $t \in T$  and  $i \in \{1, \dots, k\}$ :

$$\min_{x_{t,i}: t \in T, i \in \{1, \dots, k\}} \sum_{t,i,t',j} (\|x_{t,i} - x_{t',j}\|_2 - A_{(t,i),(t',j)})^2 \quad (3)$$

It can be solved using eigenvalue decomposition as a blackbox, that is run on a matrix which is a transformed version of  $A$ . See Borg and Groenen (2005) for more details.

### 4 Putting It All Together

Given a query of several words,  $w_1, \dots, w_k$ , we compute a two-dimensional word embedding for each word in each time slice. We then plot these embeddings through time, moving each word through the time slices using a basic line drawing algorithm based on Bresenham (1965).

The final tool we developed can be viewed online at <http://kinloch.inf.ed.ac.uk/words>.

### 5 Future Work

We hope to make this visualization tool useful for researchers who do diachronic analysis of text or words. In addition, we believe that the idea of temporal word embeddings is useful for various classification tasks, in which there is a temporal component to the data. For example, such temporal word embeddings can assist in classifying documents into the year they were written in. Preliminary work done by the first author as part of his Master's project shows that indeed this is the case, and temporal word embeddings can be used as features for a diachronic classification problem.

### Acknowledgements

We would like to thank Mirella Lapata and Bonnie Webber for useful feedback on this tool. This tool was created as part of the first author's M.Sc. project at the University of Edinburgh, School of Informatics.

### References

- [Blacoe and Lapata2012] W. Blacoe and M. Lapata. 2012. A comparison of vector-based representations for semantic composition. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 546–556. Association for Computational Linguistics.
- [Borg and Groenen2005] I. Borg and P. J. F. Groenen. 2005. *Modern multidimensional scaling: Theory and applications*. Springer.
- [Bresenham1965] J. E. Bresenham. 1965. Algorithm for computer control of a digital plotter. *IBM Systems journal*, 4(1):25–30.

- [Collobert et al.2011] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. 2011. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537.
- [Dhillon et al.2011] P. Dhillon, D. P. Foster, and L. H. Ungar. 2011. Multi-view learning of word embeddings via cca. In *Advances in Neural Information Processing Systems*, pages 199–207.
- [Faruqui and Dyer2014] M. Faruqui and C. Dyer. 2014. Improving vector space word representations using multilingual correlation. *Proc. of EACL. Association for Computational Linguistics*.
- [Firth1961] John Rupert Firth. 1961. *Papers in Linguistics 1934-1951: Repr.* Oxford University Press.
- [Michel et al.2011] J.-. Michel, Y. K. Shen, A. P. Aiden, A. Veres, M. K. Gray, J. P. Pickett, D. Hoiberg, D. Clancy, P. Norvig, J. Orwant, et al. 2011. Quantitative analysis of culture using millions of digitized books. *science*, 331(6014):176–182.
- [Mikolov et al.2011] Tomas Mikolov, Stefan Kombrink, Anoop Deoras, Lukar Burget, and J Cernocky. 2011. Rnnlm-recurrent neural network language modeling toolkit. In *Proc. of the 2011 ASRU Workshop*, pages 196–201.
- [Mnih and Hinton2007] A. Mnih and G. Hinton. 2007. Three new graphical models for statistical language modelling. In *Proceedings of the 24th international conference on Machine learning*, pages 641–648. ACM.
- [Turian et al.2010] J. Turian, L. Ratinov, and Y. Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics.